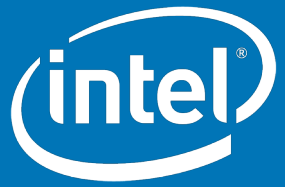

Potential Show-Stoppers for Transactional Synchronization

PPoPP '07 Panel Session

Ali-Reza Adl-Tabatabai

Programming Systems Lab

Intel Corporation

Killing the Feng Shui

TM promised to bring harmony

- Programmer declares atomicity
- System implements under the hood

But we made compromises

- Lock-free → lock-based
- Isolation & memory ordering
- Explicit locking & compensating actions
- Explicit function annotations
- Virtualized HW TM → HW acceleration

And we've only just begun...

More challenges remain

- Language & library integration
- Handling I/O
- Nested parallelism
- Communication
- Handling legacy code
- Real applications & large transactions
- Contention management
- Performance predictability
- Single thread overheads
- Performance & debug tools
- External transaction managers
- . . . I probably forgot something

Will transactions provide enough value when we're done?

The brighter side

- Databases have used transactions successfully for years
  - There's more we can learn here
- New languages supporting transactions from ground up
  - Fixes some of the warts
- TM HW has other uses
  - Speculative threading
  - Speculative optimizations
  - Speculative lock elision
- STMs might enable new features
  - Debugging

Parting thoughts

- We've compromised some of TM's elegance
- More research challenges remain
- Will it provide enough value over locks when we're done?
- Under promise so we can (over) deliver